Amendments to the Claims

Please amend Claims 1, 2, 5, 9, and 13. The Claim Listing below will replace all prior versions of the claims in the application:

Claim Listing

1. (Currently amended) A method for updating a lookup table comprising the steps of:

providing access to a first set of routes stored in nodes of a first subtree within a tree, the first subtree being accessed through a first pointer to a first subtree root node;

storing a second set of routes ~~stored~~ in nodes of a second subtree separate from the tree, the second subtree being accessed through a second pointer to a second subtree root node, while access is provided to the first set of routes stored in the first subtree by the first pointer; and

switching access to the second set of routes stored in the second subtree by replacing the first pointer to the first subtree root node with the second pointer to the second subtree root node to update the tree by replacing the first subtree with the second subtree.

2. (Currently amended) The method as claimed in Claim 1 further comprising the step of:

deallocating memory used by the first set of routes after switching access.

3. (Original) The method as claimed in Claim 1 wherein the number of routes in the first set of routes is less than the number of routes in the second set of routes.

4. (Original) The method as claimed in Claim 1 wherein the number of routes in the first set of routes is greater than the number of routes in the second set of routes.

5. (Currently amended) An apparatus for updating a lookup table comprising:

a first pointer to a first subtree root node, the first subtree root node providing access to a first set of routes stored in nodes of a first subtree within a tree; and

a second pointer to a second subtree root node, the second subtree root node providing access to a second set of routes stored in nodes of a second subtree separate from the tree, while access is provided to the first set of routes by the first pointer and switching access to the second set of routes by replacing the first pointer to the first subtree root node with the second pointer to the second subtree root node to update the tree by replacing the first subtree with the second subtree.

6. (Previously presented) The apparatus as claimed in Claim 5 further comprising:

means for deallocating memory used by the first set of routes after switching access.

7. (Original) The apparatus as claimed in Claim 5 wherein the number of routes in the first set of routes is less than the number of routes in the second set of routes.

8. (Original) The apparatus as claimed in Claim 5 wherein the number of routes in the first set of routes is greater than the number of routes in the second set of routes.

9. (Currently amended) An apparatus for updating a lookup table comprising:

a first pointer to a first subtree root node, the first subtree root node providing access to a first set of routes stored in nodes of a first subtree within a tree;

a second set of routes stored in nodes of a second subtree separate from the tree, the second subtree being accessed by a second pointer, while access is provided to the first set of routes stored in the first subtree by the first pointer; and

logic which provides access to the second set of routes by replacing the first pointer to the first subtree root node with the second pointer to a second subtree root node to update the tree by replacing the first subtree with the second subtree.

10. (Previously presented) The apparatus as claimed in Claim 9 further comprising:

deallocation logic which deallocates memory used by the first set of routes after the first pointer is replaced.

11. (Original) The apparatus as claimed in Claim 9 wherein the number of routes in the first set of routes is less than the number of routes in the second set of routes.

12. (Original) The apparatus as claimed in Claim 9 wherein the number of routes in the first set of routes is greater than the number of routes in the second set of routes.

13. (Currently amended) A method for updating a lookup table, the lookup table providing a longest prefix match for a destination address, comprising the steps of:

providing access to a first set of routes stored in nodes of a first subtree within a tree, the first subtree being accessed through a first pointer to a first subtree root node;

storing a second set of routes ~~stored~~ in nodes of a second subtree separate from the tree, the second subtree being accessed through a second pointer to a second subtree root node, while access is provided to the first set of routes stored in the first subtree by the first pointer; and

switching access to the second set of routes stored in the second subtree by replacing the first pointer to the first subtree root node with the second pointer to the second subtree root node to update the tree by replacing the first subtree with the second subtree.

14. (Previously presented) The method of claim 13, wherein the first set of routes and the second set of routes include a longest prefix route for the destination address.

15. (Previously presented) The method of claim 14, wherein the destination address includes an Internet Protocol address.

16. (Previously presented) The method of claim 14, wherein the second set of routes includes another route corresponding to the longest prefix route for another destination address.

17. (Previously presented) The method of claim 13, wherein the first set of routes and the second set of routes are associated with nodes at the bottom level of a subtree.